On Privacy-Preserving Biometric Authentication

Aysajan Abidin

KU Leuven – COSIC and iMinds, Leuven, Belgium
aysajan.abidin@esat.kuleuven.be

Abstract. Biometric authentication is becoming increasingly popular as a convenient authentication method. However, the privacy and security issues associated with biometric authentication are very serious. Privacy-preserving biometric authentication addresses privacy concerns associated with the use of biometrics and offers a secure solution for user authentication. Given the tremendous expansion of wireless communications a new distributed architecture in biometric authentication is evolving. In this distributed setting, a resource constrained client may outsource part of the computations during the biometric authentication process to a more powerful device (cloud server). In this work, we consider one such distributed setting consisting of clients, a cloud server, and a service provider and make a case for the need for verifiable computation to achieve security against malicious, as opposed to an honest-but-curious, cloud server. In particular, we propose to use verifiable computation on top of an homomorphic encryption scheme to verify that the cloud server correctly performs the computations outsourced to it. A proof of security of a generic protocol in the presence of a malicious cloud server is also provided. Finally, we discuss how an XOR-linear message authentication code can be used to verify the correctness of the computation.

Key words: Biometric authentication, biometric template privacy, homomorphic encryption, verifiable computation, XOR-linear MAC.

1 Introduction

The new era of ubiquitous computing has led to mobile biometric authentication in which resource constrained devices are involved in the authentication process. More precisely, in this setting the client gains access to the authentication system via a wireless resource constrained device (e.g., mobile phone) and part of computations involved in the authentication process are outsourced to more powerful devices (cloud servers). Although this distributed setting seems to be quite natural given the tremendous expansion of wireless communications and cloud computing, it also poses serious security and privacy concerns, since biometrics may reveal sensitive private information and could be used to profile and track individuals. In order to protect against such privacy threats, it is

important to employ privacy-preserving techniques suitable for distributed settings such as secure multi-party computation techniques.

By adopting a distributed model of internal entities in the biometric authentication process one can limit the amount of power each single protocol entity has at its disposal and consequently avoid single point of failure attacks [1]. Additionally, such separation of protocol entities ensures higher degree of privacy for the biometric data since not a single entity has access to all sensitive data (i.e., fresh biometric template, stored biometric template, user's identity). However, an important problem that rises when part of the computations of the biometric authentication process are outsourced to cloud servers is how to guarantee the confidentiality of the outsourced data as well as the correctness of the outsourced computation. A malicious cloud server could indeed modify the process in order to gain some advantages, for instance, to reduce the cost of computation or recover private information. In this paper, we treat such cases of malicious cloud server and make a case for the need for combining privacy-preserving biometric authentication with verifiable delegation of computation to protect the privacy of the biometric templates against the cloud.

Biometric authentication comprises of two phases: the *enrollment* phase and the *authentication* phase. In the enrollment phase, users provide their biometric templates derived from their biometrics (such as fingerprints, face recognition and iris scan) for storage in a database. In the *authentication* phase, users authenticate themselves by providing their fresh biometric templates, and they are authenticated if their fresh biometric template matches the reference biometric template stored in the database.

Following the previous work by [2, 3], we consider the following setting for a biometric authentication system comprising three entities, namely, a client set $\mathcal{C}$ of clients $\mathcal{C}_i$, for $i = 1, \cdots, N$, one for each user $\mathcal{U}_i$, a computation (or a cloud) server $\mathcal{CS}$ with a database $\mathcal{DB}$, and a service provider $\mathcal{SP}$. The client $\mathcal{C}_i$ has a sensor that captures biometric templates from its owner (i.e., the user $\mathcal{U}_i$). The cloud server $\mathcal{CS}$ stores the reference biometric templates and performs computationally expensive calculations. The service provider $\mathcal{SP}$ takes the final decision depending on whether there is a match between the fresh and the reference biometric templates. This is a reasonable model considering the fast rise of cloud computing and storage services, and also the widespread use of smartphones with embedded biometric sensors.

A common cryptographic tool that is employed in building privacy-preserving biometric authentication is homomorphic encryption [1–7]. In such a scheme, encryption protects the privacy of the biometric templates

while the matching of the fresh and reference biometric templates are performed over the encrypted data using the homomorphic property of the encryption. However, this requires the actor responsible for performing the delegated calculations on encrypted biometric templates to be trusted. Otherwise, by computing a function different than what the protocol specifies and using $\mathcal{SP}$ as an oracle, the computing actor (i.e., the $\mathcal{CS}$) can learn information about either the stored reference biometric template b_i or the fresh biometric template b'_i. Similar attacks on two recently proposed protocols employing ring-LWE and ideal lattice based somewhat homomorphic encryption schemes [2, 3] are presented in [8]. Therefore, in addition to homomorphic encryption, a cryptographic scheme that allows the client/service provider to verify that the cloud server performed the correct computation. Schemes that allow verification of computations delegated to a computationally powerful third party (or the cloud server in our case) already exist and are known as verifiable computation [9–14] or signatures of correct computation [15]. In this paper, we study their employment in privacy-preserving biometric authentication.

1.1 Related work.

Over the years, quite a few proposals for privacy-preserving biometric authentication appeared in the literature. These are based upon cryptographic techniques, such as blivious transfer [16, 17], private information retrieval [18, 19], and homomorphic encryption [20, 21]. For example, Bringer *et al.* employed the Goldwasser-Micali cryptosystem [21] to protect the privacy of the biometric templates against *honest-but-curious* (or *passive*) adversaries in [1]. There are also other privacy-preserving biometric authentication protocols that are based on the additive HE by Paillier [20] and Damgård *et al.* [22] such as the protocols for face recognition in [5–7]. Oblivious transfer was used in SCiFi [23], a system for secure computation of face identification. Furthermore, somewhat HE schemes based on ideal lattices and ring learning with errors are also employed in designing privacy-preserving biometric authentication protocols in [2, 3].

All of these protocols are designed to be secure against *honest-but-curious* adversaries, and their security and privacy properties are later analysed in [4, 8, 24–26]. In [24], Simoens *et al.* made a compelling case for the need for designing privacy-preserving biometric authentication protocols that are secure against *malicious* adversaries. They also presented a framework for analysing the security and privacy-preserving properties of biometric authentication protocols in the presence of such adversaries. In fact, the weaknesses of the protocols proposed in [1–3] that are identified

in [8, 25, 26] can be attributed to the lack of verifiable computation. In other words, the attacks reported in [8, 25, 26] can also be mitigated using verifiable computation.

Since most biometric authentication schemes use binary biometric templates, the Hamming distance (or the normalised Hamming distance) is employed to check whether two biometric templates match each other. Therefore, protocols for secure Hamming distance computation based on oblivious transfer are proposed by Bringer, Chabanne and Patey in [27]. These protocols have potential applications in privacy-preserving biometric authentication. Recently Bringer *et al.* generalised their results for secure computation of other distances such as the Euclidean and the normalised Hamming distance in [28].

1.2 Our contribution.

In this paper, we propose to combine verifiable computation with homomorphic encryption in order to achieve security against malicious computing server (i.e., the cloud server) in the above mentioned distributed biometric authentication setting. To this end, we outline a generic biometric authentication protocol with enhanced security and privacy properties in the presence of a malicious cloud server, combining homomorphic encryption with a scheme for verifiable computation. We then prove the security of the generic protocol against malicious cloud server. Furthermore, we discuss how an XOR-linear message authentication code (MAC) can be used to verify the correctness of the outsourced computation in the studied biometric authentication setting.

Outline. The rest of the paper is organised as follows. Section 2 introduces the necessary background. Section 3 presents our threat model and communication model for the protocol. Next we propose a generic protocol combining a scheme for verifiable computation with HE, and show that the protocol has enhanced security and privacy properties even in the presence of a malicious cloud server in Section 4. Furthermore, we give a specific instantiation of our generic protocol using an XOR-linear MAC in Section 5. Finally, Section 6 concludes the paper.

2 Preliminaries

First, we introduce the notations used in this paper. Biometric templates are regarded as vectors in $\mathbb{Z}_{q\geqslant 2}^N$, where q is an integer. Let b_i and b_i'

denote the reference and fresh biometric templates, respectively, of the i-th user $\mathcal{U}_i$ whose identity is denoted by ID_i, for $i = 1, \cdots, M$, where M is the total number of users. Let τ be the authentication threshold and $\mathsf{Dist} : \mathbb{Z}_q^M \times \mathbb{Z}_q^M \mapsto \mathbb{R}_{\geqslant 0}$ be a distance on $\mathbb{Z}_q^M$. Then we say that b_i and b_i' match each other and thus belong to the same user, if $\mathsf{Dist}(b_i, b_i') \leqslant \tau$. In the case of binary templates, the Hamming distance between b_i and b_i' is denoted by $\mathsf{HD}(b_i, b_i')$, which is also equal to the Hamming weight $\mathsf{HW}(b_i \oplus b_i')$. Finally, PPT and IND-CPA refer to probabilistic polynomial time and indistinguishability against chosen plaintext attacks, respectively.

2.1 Homomorphic encryption

We use an homomorphic encryption (HE) scheme, denoted by $\mathsf{HE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$, that allows, given $\mathsf{Enc}(b_i)$ and $\mathsf{Enc}(b_i')$, to compute $\mathsf{Enc}(\mathsf{Dist}(b_i, b_i'))$ homomorphically. We require the employed HE scheme to have semantic security against chosen plaintext attacks, which is defined as follows. Let $(\mathsf{pk}, \mathsf{sk})$ be the public and private key pairs for the HE scheme and λ a security parameter. Consider the following game played between a PPT adversary $\mathcal{A}$ and a challenger

$\mathsf{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathsf{HE},\mathcal{A}}(\lambda)$:
$(\mathsf{pk}, \mathsf{sk}), \leftarrow \mathsf{KeyGen}(\lambda)$; $(m_0, m_1),\ m_0 \neq m_1 \leftarrow \mathcal{A}(\lambda, \mathsf{pk})$; $\beta \xleftarrow{R} \{0,1\}$
$c \leftarrow \mathsf{Enc}(m_\beta, \mathsf{pk})$; $\beta' \leftarrow \mathcal{A}(m_0, m_1, c, \mathsf{pk})$
Return 1 if $\beta' = \beta$, 0 otherwise

and define the adversary's advantage in this game as $\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{HE},\mathcal{A}}(\lambda) = \left|2 \Pr\left\{\mathsf{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathsf{HE},\mathcal{A}}(\lambda) = 1\right\} - 1\right|$.

Definition 1. *We say that HE is IND-CPA-secure if all PPT adversaries have a negligible advantage in the above game:* $\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{HE},\mathcal{A}}(\lambda) \leqslant \mathsf{negl}(\lambda)$.

Here, $\mathsf{negl}(\lambda)$ is a negligible function defined as follows.

Definition 2. *We say that a function* $\mathsf{negl} : \mathbb{N} \mapsto [0,1]$ *is negligible if for all positive polynomials* poly *and all sufficiently large* $\lambda \in \mathbb{N}$*, we have* $\mathsf{negl}(\lambda) < 1/\mathsf{poly}(\lambda)$.

2.2 Privacy-preserving biometric authentication

At a high level, a privacy-preserving biometric authentication (PPBA) protocol employing HE can be defined by the following processes.

- Setup: In this step, the keys $(\mathsf{pk}, \mathsf{sk})$ for the HE scheme are generated and distributed to the relevant protocol actors by either one protocol actor or an external trusted third party.

- $\mathcal{DB} \leftarrow \mathsf{Enroll}\big((\mathsf{Enc}(b_i))_{i=1}^{M}, (\mathsf{ID}_i)_{i=1}^{M}\big)$: This process collects the encrypted reference biometric template $\mathsf{Enc}(b_i)$ and identity ID_i pair from all M users and stores them in the database $\mathcal{DB}$.
- $1 \cup 0 \leftarrow \mathsf{Authen}(\mathsf{Enc}(b'_i), \mathsf{ID}_i)$: To authenticate a user $\mathcal{U}_i$, this process takes an encrypted fresh biometric template $\mathsf{Enc}(b'_i)$ and a claimed identity ID_i, retrieves $\mathsf{Enc}(b_i)$ from the database $\mathcal{DB}$, and homomorphically computes $\mathsf{Dist}(b_i, b'_i)$ from $\mathsf{Enc}(b_i)$ and $\mathsf{Enc}(b'_i)$. Finally, it outputs 1 if the authentication is successful, 0 otherwise.

A PPBA protocol must be both correct and secure.

Definition 3. *We say that a PPBA protocol is correct if, for all enrolled user identities ID_i with the corresponding reference biometric templates b_i, and for all fresh biometric templates b'_i with $\mathsf{Dist}(b_i, b'_i) \leqslant \tau$, it is always the case that $1 \leftarrow \mathsf{Authen}(\mathsf{Enc}(b'_i), \mathsf{ID}_i)$.*

One may argue that one can set the Authen process to always return 1 and thus violate the correctness. However, the Authen process described here is just an abstraction for the verification process of a biometric authentication protocol, so for it to return 1, the fresh biometric template must match the reference biometric template.

Informally, a PPBA protocol is secure if a malicious adversary, which in our case is the cloud server, cannot learn more about the biometric templates than what is already revealed by the protocol transcripts. Formally, we define the security of against a malicious adversary $\mathcal{A}$ as follows. Consider the following game

$\mathsf{Exp}^{\mathsf{Priv}}_{\mathsf{PPBA},\mathcal{A}}(\lambda)$:
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\lambda)$; $(\mathsf{ID}_i, b'_{i_0}, b'_{i_1}), b'_{i_0} \neq b'_{i_1} \leftarrow \mathcal{A}(\lambda, \mathsf{pk})$
$\beta \xleftarrow{R} \{0,1\}$; $\mathsf{Out} \leftarrow \mathsf{Authen}\big(\mathsf{ID}_i, \mathsf{Enc}(b'_{i_\beta})\big)$
$\beta' \leftarrow \mathcal{A}\big(\mathsf{ID}_i, b'_{i_0}, b'_{i_1}, \mathsf{Enc}(b'_{i_\beta}, \mathcal{DB}), \mathsf{Out}\big)$
Return 1 if $\beta' = \beta$, 0 otherwise

and define the adversary's advantage in this game as $Adv^{\mathsf{Priv}}_{\mathsf{PPBA},\mathcal{A}}(\lambda) = \big|2\Pr\{\mathsf{Exp}^{\mathsf{Priv}}_{\mathsf{PPBA},\mathcal{A}}(\lambda) = 1\} - 1\big|$. Note, ID_i has to be an enrolled user identity.

Definition 4. *We say that a PPBA protocol is secure if all PPT adversaries have a negligible advantage in the above game: $Adv^{\mathsf{Priv}}_{\mathsf{PPBA},\mathcal{A}}(\lambda) \leqslant \mathsf{negl}(\lambda)$.*

We assume that the adversary is given an oracle access to Authen and is allowed to query it with user $\mathsf{ID}_j (\neq \mathsf{ID}_i)$ and b'_j polynomially many times (e.g., $\mathsf{poly}(\lambda)$ times). The adversary is also given $\mathsf{Enc}(b'_{i_\beta})$ and the

database. If the adversary cannot distinguish whether it is $(\mathsf{ID}_i, b'_{i_0})$ or $(\mathsf{ID}_i, b'_{i_1})$ that is being used by Authen, then we say that the protocol preserves privacy of the biometric templates.

2.3 Verifiable computation

A scheme for verifiable computation (VC) allows a computationally weak client to both outsource heavy computations to a computationally powerful cloud server and efficiently verify the output of the cloud server. In our case, we consider that the heavy computations outsourced by the client to the cloud server are performed over encrypted data. In particular, the cloud computes a function f on input $\mathsf{Enc}(b_i)$ and $\mathsf{Enc}(b'_i)$ so that $f(\mathsf{Enc}(b_i), \mathsf{Enc}(b'_i)) = \mathsf{Enc}(\mathsf{Dist}(b_i, b'_i))$.

Definition 5 (Verifiable computation [11]). *A VC scheme* $\mathsf{VC} = (\mathsf{KeyGen}, \mathsf{ProbGen}, \mathsf{Com}, \mathsf{Ver})$ *comprises four algorithms defined as:*

- $(\mathsf{PK}, \mathsf{VK}) \leftarrow \mathsf{KeyGen}(\lambda, f)$*: The (randomised) key generation algorithm* KeyGen *takes as input a security parameter* λ *and a function* f*, and outputs a public key* PK *and a verification key* VK *for the function* f*. The public key* PK *is provided to the cloud server, while the verification key* VK *is kept secret by the client.*
- $(\sigma_x, \rho_x) \leftarrow \mathsf{ProbGen}(x, \mathsf{VK})$*: The problem generation algorithm* $\mathsf{ProbGen}$ *takes as input a function input* x *and a verification key* VK*, and outputs a public value* σ_x *and a secret value* ρ_x*. The public value* σ_x *is provided to the cloud, while the secret value* ρ_x *is kept secret by the client.*
- $\sigma_y \leftarrow \mathsf{Com}(\sigma_x, \mathsf{PK})$*: The computation algorithm* Com *takes as input a public value* σ_x *and a public key* PK *for* f*, and outputs an encoded version* σ_y *of* $y = f(x)$*.*
- $y \cup \perp \leftarrow \mathsf{Ver}(\rho_x, \sigma_y, \mathsf{VK})$*: The verification algorithm* Ver *takes as input a verification key* VK*, a secret value* ρ_x*, and the output from* Com*, and outputs* y *indicating that* σ_y *is a valid encoding of* $y = f(x)$ *or* $\perp$ *indicating that* σ_y *does not represent* $f(x)$*.*

A VC scheme is correct if the output of the problem generation algorithm $\mathsf{ProbGen}$ allows an honest cloud server to compute values that will be successfully verified and that correspond to the evaluation of f on the input values. Formally, correctness is defined as follows.

Definition 6. *A VC scheme* VC *is said to be correct if, for any function* f *and input* x *in the domain of* f*, it holds that* $y \leftarrow \mathsf{Ver}(\rho_x, \sigma_y, \mathsf{VK})$ *as long as* $(\mathsf{PK}, \mathsf{VK}) \leftarrow \mathsf{KeyGen}(\lambda, f)$*,* $(\sigma_x, \rho_x) \leftarrow \mathsf{ProbGen}(x, \mathsf{VK})$*, and* $\sigma_y \leftarrow \mathsf{Com}(\sigma_x, \mathsf{PK})$*.*

In order to be secure, a VC scheme VC must be such that, for any given function f and input x, a malicious cloud should not be able to make the verification algorithm accept y' such that $y' \neq f(x)$. Formally, the security of VC is defined as the advantage of an adversary in the following game $\mathsf{Exp}_{\mathsf{VC},\mathcal{A}}(\lambda, f)$ which captures the intuitive argument above.

$\mathsf{Exp}_{\mathsf{VC},\mathcal{A}}(\lambda, f)$:
$(\mathsf{PK}, \mathsf{VK}) \leftarrow \mathsf{KeyGen}(\lambda, f)$
$x_1 \leftarrow \mathcal{A}(\lambda, \mathsf{PK})$
$(\sigma_{x_1}, \rho_{x_1}) \leftarrow \mathsf{ProbGen}(x_1, \mathsf{VK})$
$\sigma_{y_1} \leftarrow \mathcal{A}(\mathsf{PK}, x_1, \sigma_{x_1})$
$\beta_1 \leftarrow \mathsf{Ver}(\rho_{x_1}, \sigma_{y_1}, \mathsf{VK})$
For $i = 2, \cdots, \ell = \mathsf{poly}(\lambda)$
  $x_i \leftarrow \mathcal{A}(\mathsf{PK}, x_1, \sigma_{x_1}, \beta_1, \cdots, x_{i-1}, \sigma_{x_{i-1}}, \beta_{i-1})$
  $(\sigma_{x_i}, \rho_{x_i}) \leftarrow \mathsf{ProbGen}(x_i, \mathsf{VK})$
  $\sigma_{y_i} \leftarrow \mathcal{A}(\mathsf{PK}, x_1, \sigma_{x_1}, \beta_1, \cdots, x_{i-1}, \sigma_{x_{i-1}}, \beta_{i-1}, \sigma_{x_i})$
  $\beta_i \leftarrow \mathsf{Ver}(\rho_{x_i}, \sigma_{y_i}, \mathsf{VK})$
$x \leftarrow \mathcal{A}(\mathsf{PK}, x_1, \sigma_{x_1}, \beta_1, \cdots, x_\ell, \sigma_{x_\ell}, \beta_\ell)$
$(\sigma_x, \rho_x) \leftarrow \mathsf{ProbGen}(x, \mathsf{VK})$
$\sigma'_y \leftarrow \mathcal{A}(\mathsf{PK}, x_1, \sigma_{x_1}, \beta_1, \cdots, x_\ell, \sigma_{x_\ell}, \beta_\ell, \sigma_x)$
$y' \leftarrow \mathsf{Ver}(\rho_x, \sigma'_y, \mathsf{VK})$
Return 1 if $y' \neq f(x)$ and $y' \neq \perp$, 0 otherwise.

The adversary's advantage in this game is defined as $\mathsf{Adv}_{\mathsf{VC},\mathcal{A}}(\lambda, f) = \Pr\left\{\mathsf{Exp}_{\mathsf{VC},\mathcal{A}}(\lambda, f) = 1\right\}$. Note that the adversary is given an oracle access to $\mathsf{ProbGen}$ and Ver.

Definition 7 (Security of VC [11]). *We say that VC is secure if, for any function f, all PPT adversaries have a negligible advantage in the above game: $\mathsf{Adv}_{\mathsf{VC},\mathcal{A}}(\lambda, f) \leqslant \mathsf{negl}(\lambda)$.*

3 Threat model

When analysing the security of a protocol, there are two types of adversaries to consider: a *semi-honest* (also known as, *honest-but-curious* or *passive*) adversary and a *malicious* (or *active*) adversary. A *semi-honest* adversary follows the protocol correctly, but attempts to deduce as much information as possible about protected data from the protocol transcripts. A *malicious* adversary, on the other hand, can arbitrarily deviate from the protocol specifications. Both types of adversaries attempt to break either the correctness or the security property of the protocol. Here we focus on *malicious* adversaries.

We consider a three-party setting which comprises a client $\mathcal{C}_i$ (one for each user $\mathcal{U}_i$), a cloud server $\mathcal{CS}$, and a service provider $\mathcal{SP}$. The client $\mathcal{C}_i$ (e.g., a smartphone owned by the user $\mathcal{U}_i$) has a biometric sensor that extracts biometric templates from the user. We assume that each user's

client device is not compromised. Since if a client $\mathcal{C}_i$ is compromised, then the reference biometric template of the owner $\mathcal{U}_i$ can be easily recovered using the fresh biometric template provided by the owner [29]. The service provider $\mathcal{SP}$ manages the keys for the employed encryption scheme and makes the authentication decision. Therefore, we consider the service provider $\mathcal{SP}$ as a trusted protocol actor. However, we do not entrust any biometric template to the service provider. The malicious actor is the cloud server $\mathcal{CS}$, which has a database storing the encrypted reference biometric templates and performs computations on the encrypted fresh and reference biometric templates. The result of the computation performed by $\mathcal{CS}$ will allow $\mathcal{SP}$ to make its decision. In this paper, we exclusively focus on biometric template privacy and template recovery attacks. Hence, denial-of-service type of attacks are outside the scope of this paper.

For the communication model, we assume that the communication channel between the protocol entities are both authentic and secure in the sense that messages exchanged between two parties cannot be modified or intercepted by an eavesdropper. This assumption is also necessary for avoiding replay attacks. Such a communication channel can be established by using TLS or IPsec between the protocol participants.

4 A generic protocol

This section presents a generic protocol that combines verifiable computation with an homomorphic encryption. The protocol also employs a collision resistant cryptographic hash function $H : \{0,1\}^\star \mapsto \{0,1\}^n$ (in our security analysis, we regard H as a random oracle). To differentiate from the database $\mathcal{DB}$ on the cloud server side, we use db to denote the database on the service provider side. We call the generic protocol PPBA which comprises the following.

- Enroll: The user enrollment phase is depicted in Fig. 1. The service provider $\mathcal{SP}$ chooses a collision resistant cryptographic hash function H and runs the key generation algorithm KeyGen for the HE and VC schemes using a security parameter λ and the function f to be computed by the cloud as input: $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{HE.KeyGen}(\lambda)$ and $(\mathsf{PK}, \mathsf{VK}) \leftarrow \mathsf{VC.KeyGen}(\lambda, f)$. The client $\mathcal{C}_i$ requests enrollment by sending its owner $\mathcal{U}_i$'s identity ID_i to $\mathcal{SP}$. $\mathcal{SP}$ then maps ID_i to an index i using a process known only to itself. The tuple (i, H, pk, VK) is sent to $\mathcal{C}_i$, and (pk, PK) to $\mathcal{CS}$. The function f is known to the protocol actors. After receiving (i, H, pk), $\mathcal{C}_i$ first obtains the reference

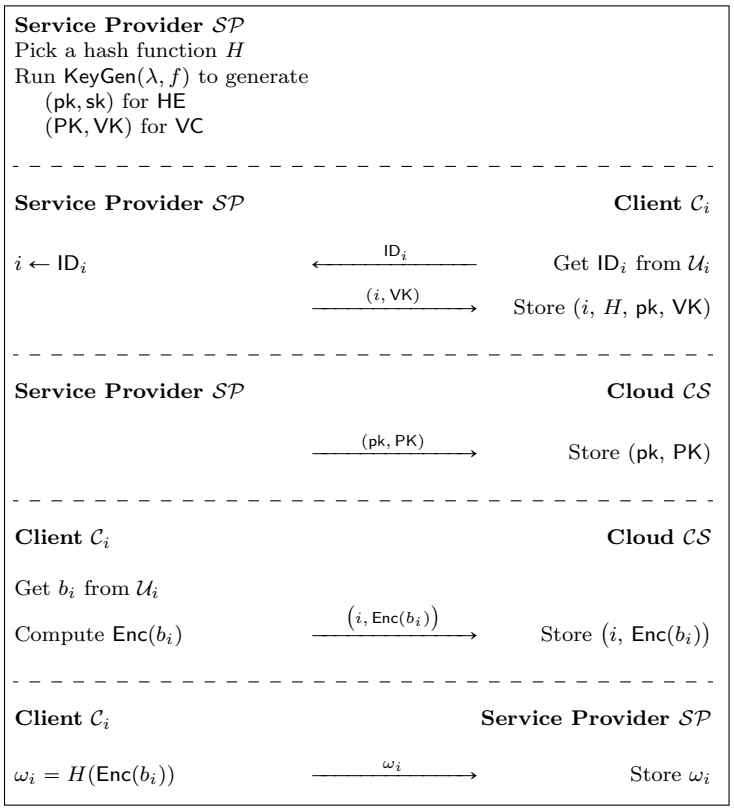

Fig. 1: The enrollment phase of PPBA.

biometric template b_i and encrypts the reference biometric template, $\mathsf{Enc}(b_i)$. $\mathcal{C}_i$ then provides $(i, \mathsf{Enc}(b_i))$ to the database $\mathcal{DB}$ on the cloud server side for storage. In addition, $\mathcal{C}_i$ sends the hash $\omega_i = H(\mathsf{Enc}(b_i))$ to $\mathcal{SP}$ which stores (i, ω_i) in its database db. Locally, $\mathcal{C}_i$ stores (i, VK). Since it is necessary for security, we assume that user enrollment is performed in a secure and controlled environment.

– Authen: In this phase, before the user $\mathcal{U}_i$ authenticates himself, the service provider $\mathcal{SP}$ authenticates itself to the client $\mathcal{C}_i$ and provides the public key pk for HE and the hash function H to $\mathcal{C}_i$. The authentication of $\mathcal{SP}$ is necessary to avoid sending sensitive information to a malicious party impersonating the legitimate $\mathcal{SP}$. After $\mathcal{SP}$ is authenticated, $\mathcal{C}_i$ obtains from its user $\mathcal{U}_i$ a fresh biometric template b'_i and an identity ID_i, and provides $\mathsf{Enc}(b'_i)$ and the index i that it stored during enrollment to the cloud server $\mathcal{CS}$. The cloud then re-

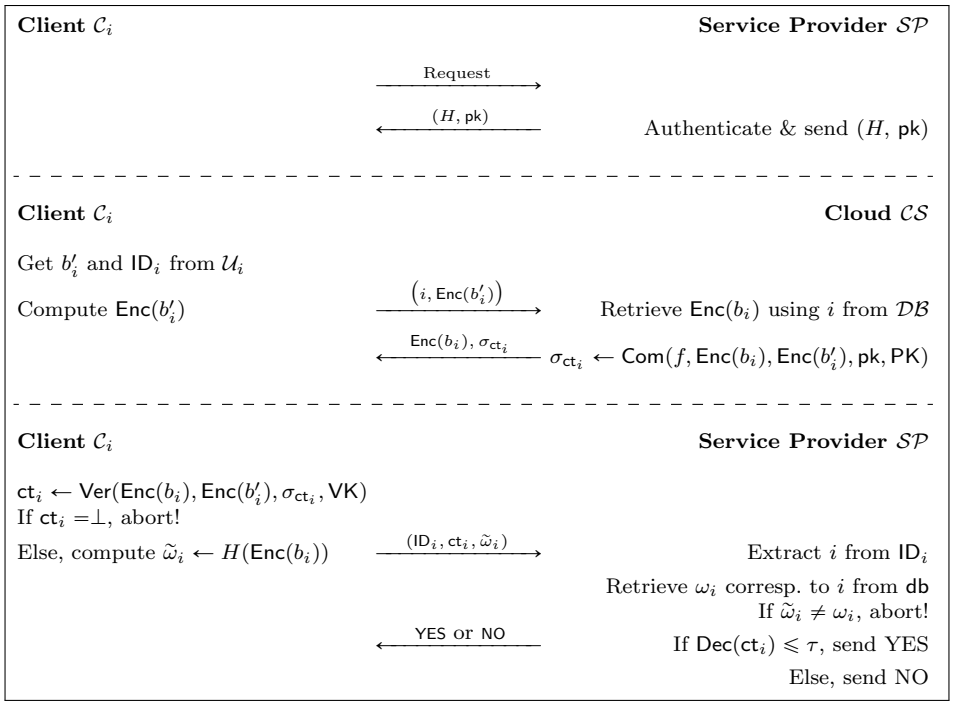

Fig. 2: The user authentication phase of PPBA.

trieves $\mathsf{Enc}(b_i)$ corresponding to i from its database $\mathcal{DB}$ and runs the computation algorithm $\sigma_{\mathsf{ct}_i} \leftarrow \mathsf{Com}(\mathsf{Enc}(b_i), \mathsf{Enc}(b'_i), \mathsf{pk}, \mathsf{PK})$ for the verifiable computation scheme VC. Note that pk is needed to evaluate the function f on $\mathsf{Enc}(b_i)$ and $\mathsf{Enc}(b'_i)$. The output σ_{ct_i} is an encoded version of $\mathsf{ct}_i = f(\mathsf{Enc}(b_i), \mathsf{Enc}(b'_i)) = \mathsf{Enc}(\mathsf{Dist}(b_i, b'_i))$. Then, $\mathcal{CS}$ sends $\mathsf{Enc}(b_i)$, σ_{ct_i} back to the client $\mathcal{C}_i$, which runs the verification algorithm $\mathsf{ct}_i \leftarrow \mathsf{Ver}(\mathsf{Enc}(b_i), \mathsf{Enc}(b'_i), \sigma_{\mathsf{ct}_i}, \mathsf{VK})$. If $\mathsf{ct}_i \neq \perp$, then $\mathcal{C}_i$ computes $\widetilde{\omega}_i = H(\mathsf{Enc}(b_i))$ and sends $(\mathsf{ID}_i, \mathsf{ct}_i, \widetilde{\omega}_i)$ to $\mathcal{SP}$; otherwise, $\mathcal{C}_i$ aborts the protocol. Upon receiving $(\mathsf{ID}_i, \mathsf{ct}_i, \widetilde{\omega}_i)$ from $\mathcal{C}_i$, $\mathcal{SP}$ first extracts i from ID_i, retrieves ω_i from db and checks whether $\widetilde{\omega}_i = \omega_i$. Note here that the hash function is used to check whether the cloud used the correct input, i.e., $\mathsf{Enc}(b_i)$, to the function f. If $\widetilde{\omega}_i = \omega_i$, then $\mathcal{SP}$ decrypts ct_i, i.e., $\mathsf{Dec}(\mathsf{ct}_i) = \mathsf{Dec}\big(\mathsf{Enc}(\mathsf{Dist}(b_i, b'_i))\big) = \mathsf{Dist}(b_i, b'_i)$. If $\mathsf{Dist}(b_i, b'_i) \leqslant \tau$, then it outputs 1 (or YES) meaning that the client $\mathcal{C}_i$ (or the user $\mathcal{U}_i$) is authenticated; otherwise, it outputs 0 (or NO) meaning that the client $\mathcal{C}_i$ (or the user $\mathcal{U}_i$) is not authenticated.

Remark 1: We note that the problem generation algorithm $\mathsf{ProbGen}$ for the VC scheme is not used above since in our case the public and secret output of $\mathsf{ProbGen}$ algorithm are the same and equal to $(\mathsf{Enc}(b_i), \mathsf{Enc}(b'_i))$.

Remark 2: By requiring the correspondence between an identity and an index (e.g., $\mathsf{ID}_i \leftrightarrow i$) to be known only to the service provider, we can prevent a potentially malicious client $\mathcal{C}_i$ from impersonating another client $\mathcal{C}_j$, $j \neq i$. If this is not the case, then a misbehaving client, say $\mathcal{C}_i$, can initiate the authentication phase with an identity ID_j, $j \neq i$, and index j and obtain $\mathsf{Enc}(b_j)$ from the cloud $\mathcal{CS}$. Then, $\mathcal{C}_i$ aborts the current round and later authenticates itself as ID_j using $\mathsf{Enc}(b_j)$. Note that this also guarantees identity privacy against since $\mathcal{CS}$ does not know to which user identity a database entry belongs.

It is straightforward to see that the correctness of the generic protocol readily follows. The following theorem summarises the security of the generic protocol PPBA against the malicious cloud server. The proof of the theorem is given in Appendix A.

Theorem 1 (Security of PPBA). *Let H be a random oracle. Let HE be an $\mathsf{IND\text{-}CPA}$-secure HE scheme and VC a secure VC scheme as defined in Definition 7. Let $\mathcal{A}$ be a malicious cloud server that is PPT. Then the advantage of $\mathcal{A}$ in the game $\mathsf{Exp}^{Priv}_{PPBA,\mathcal{A}}(\lambda, f)$ (cf. Section 2.1) is negligible, i.e., $\mathsf{Adv}_{PPBA,\mathcal{A}}(\lambda, f) \leqslant \mathsf{negl}(\lambda)$.*

As mentioned in the previous work, the protocols previously proposed in [1–3] can be enhanced with a suitable verifiable computation scheme to mitigate the reported attacks in [8, 25, 26].

5 Instantiation

Here we discuss an instantiation of the generic protocol using an $\oplus$-linear message authentication code (MAC), where $\oplus$ is the XOR operation.

A MAC scheme consists of three algorithms $(\mathsf{KeyGen}, \mathsf{TAG}, \mathsf{VRFY})$ (associated with a key space, a message space and a tag space). KeyGen, a key generation algorithm, takes a security parameter λ as input and outputs a key k (i.e., $\mathsf{k} \leftarrow \mathsf{KeyGen}(\lambda)$). TAG, a tag generation algorithm, takes a message m and a key k as input, and outputs a tag (i.e., $t \leftarrow \mathsf{TAG}(m, \mathsf{k})$). VRFY, a verification algorithm, takes a message m, a tag t and a key k as input, and outputs a decision $\mathsf{Out}_{\mathsf{MAC}}$ (i.e., $\mathsf{Out}_{\mathsf{MAC}} \leftarrow \mathsf{VRFY}(m, t, \mathsf{k})$), which is 1 if the message-tag pair (m, t) is valid, and 0 otherwise.

A typical construction of a MAC scheme is via the use of Universal$_2$ (U_2) hash functions, see Appendix B for definitions and how U_2 hash functions can be used to construct a MAC scheme. There are constructions of U_2 hash functions that are $\oplus$-linear [30], from which one can construct an $\oplus$-linear MAC scheme. Note that a MAC scheme is called $\oplus$-linear if $\mathsf{TAG}(m_1 \oplus m_2, \mathsf{k}) = \mathsf{TAG}(m_1, \mathsf{k}) \oplus \mathsf{TAG}(m_2, \mathsf{k})$.

Using any HE scheme that enables the evaluation of XOR of two encrypted bitstrings (e.g., the Goldwasser-Micali encryption scheme [21] which supports this) and an $\oplus$-linear MAC to verify the correctness of the computation performed by $\mathcal{CS}$, we have the following variation of the generic protocol presented in the previous section.

- Enroll: The service provider $\mathcal{SP}$ runs the key generation algorithm KeyGen for the HE and MAC schemes using a security parameter λ: $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{HE.KeyGen}(\lambda)$ and $\mathsf{k}_i \leftarrow \mathsf{MAC.KeyGen}(\lambda)$. The client $\mathcal{C}_i$ requests for enrollment by sending its owner $\mathcal{U}_i$'s identity ID_i to $\mathcal{SP}$, which then maps ID_i to an index i using a process known only to itself. The tuple (i, pk, k_i) is sent to $\mathcal{C}_i$, and pk to $\mathcal{CS}$. After receiving (i, pk, k_i), $\mathcal{C}_i$ first obtains the reference biometric template b_i and encrypts the reference biometric template, $\mathsf{Enc}(b_i)$. $\mathcal{C}_i$ then provides (i, $\mathsf{Enc}(b_i)$) to the database $\mathcal{DB}$ on the cloud server side for storage. In addition, $\mathcal{C}_i$ sends the tag $t_i = \mathsf{TAG}(b_i, \mathsf{k}_i)$ to $\mathcal{SP}$ which stores (i, k_i, t_i) in its database db. Locally, $\mathcal{C}_i$ stores (i, k_i). As before, we assume that user enrollment is performed in a secure and controlled environment.
- Authen: Again, before the user $\mathcal{U}_i$ authenticates himself, the service provider $\mathcal{SP}$ authenticates itself to the client $\mathcal{C}_i$. Then, $\mathcal{C}_i$ obtains from its user $\mathcal{U}_i$ a fresh biometric template b_i' and an identity ID_i, and provides $\mathsf{Enc}(b_i')$ and the index i to the cloud server $\mathcal{CS}$. In addition, $\mathcal{C}_i$ computes $t_i' = \mathsf{TAG}(b_i', \mathsf{k}_i)$ and sends (ID_i, t_i') to $\mathcal{SP}$. The cloud then retrieves $\mathsf{Enc}(b_i)$ corresponding to i from its database $\mathcal{DB}$ and computes $\gamma_i = \mathsf{Enc}(b_i \oplus b_i')$ homomorphically from $\mathsf{Enc}(b_i)$ and $\mathsf{Enc}(b_i')$, and sends (i, γ_i) to $\mathcal{SP}$. The service provider then extracts i from ID_i and checks if the extracted i and the index received from $\mathcal{CS}$ match each other. If they match, $\mathcal{SP}$ continues to retrieves k_i and t_i corresponding to i from db, decrypts γ_i to obtain $\widetilde{b_i \oplus b_i'}$ (i.e., $\widetilde{b_i \oplus b_i'} \leftarrow \mathsf{Dec}(\gamma_i)$), and runs the MAC verification algorithm $\mathsf{VRFY}(\widetilde{b_i \oplus b_i'}, t_i \oplus t_i', \mathsf{k}_i)$. If the output from VRFY is 0, $\mathcal{SP}$ rejects the user. Otherwise, $\mathcal{SP}$ checks if the Hamming weight $\mathsf{HW}(\widetilde{b_i \oplus b_i'}) \leqslant \tau$. Note that $\mathsf{HW}(b_i \oplus b_i') = \mathsf{HD}(b_i, b_i')$, where HD is the Hamming distance. If this is the case, $\mathcal{SP}$ authenticates the user $\mathcal{U}_i$, otherwise rejects.

5.1 Security Analysis

The instantiation is slightly different from the generic protocol in that the correctness of the computation is verified by $\mathcal{SP}$ in the instantiation, we will also present the security proof for the "instantiation" separately.

Definition 8. *A MAC scheme is called (Q_T, Q_V, t, ϵ)-secure (or, ϵ-secure, for short) if no PPT adversary $\mathcal{A}$ running in time at most t cannot generate a valid message-tag pair, even after making Q_T tag generation queries to TAG and Q_V verification queries to VRFY, except with probability ϵ.*

In any biometric template recovery attack that makes use of the side channel information (i.e., the authentication result), $\mathcal{CS}$ needs to be able to submit to $\mathcal{SP}$ a γ which encrypts a message that passes the MAC verification test performed by $\mathcal{SP}$. The ϵ-security of the employed MAC scheme does not allow this to happen. Furthermore, from a rejection response by $\mathcal{SP}$, $\mathcal{CS}$ does not know whether it is due the MAC verification failure or the mismatch between the fresh and reference biometric templates. Hence, our instantiation is robust and secure against the malicious $\mathcal{CS}$. Formally, the following summarises the security of the instantiation.

Theorem 2. *Let HE be an IND-CPA-secure HE scheme such that $\mathsf{HE.Enc}(m_1, pk)\mathsf{HE.Enc}(m_2, pk) = \mathsf{HE.Enc}(m_1 \oplus m_2, pk)$ and MAC an ϵ-secure $\oplus$-linear MAC scheme. Then, the protocol that employs the HE and MAC schemes is secure against a malicious cloud server.*

The proof is given in Appendix C.

6 Summary

Privacy-preserving biometric authentication allows to authenticate users using their biometrics while preserving the biometric privacy. A natural approach to building a privacy-preserving biometric authentication protocol is the employment of an homomorphic encryption scheme that allows the computations and the matching process over encrypted biometric data. There are indeed multiple privacy-preserving biometric authentication protocols proposed in the literature over the years that rely on homomorphic encryption (cf. Section 1.1). In this work, we proposed to combine schemes for verifiable computation with homomorphic encryption to preserve the biometric privacy in a distributed remote biometric authentication setting comprising clients, a cloud server, and a service provider. A generic biometric authentication protocol which is secure against a malicious, as opposed to honest-but-curious, cloud server is presented. Moreover,

an instantiation is also given using an XOR-linear MAC to verify the correctness of the computation performed by the cloud.

Acknowledgments. The author would like to thank the anonymous reviewers for their helpful comments. This work was supported by the European Commission through the SECURITY programme under FP7-SEC-2013-1-607049 EKSISTENZ.

References

1. Bringer, J., Chabanne, H., Izabachène, M., Pointcheval, D., Tang, Q., Zimmer, S.: An application of the Goldwasser-Micali cryptosystem to biometric authentication. In: ACISP 2007. Volume 4586 of LNCS., Springer (2007) 96–106
2. Yasuda *et al.*, M.: Packed homomorphic encryption based on ideal lattices and its application to biometrics. In: Security Engineering and Intelligence Informatics. Volume 8128 of LNCS. (2013) 55–74
3. Yasuda *et al.*, M.: Practical packing method in somewhat homomorphic encryption. In: DPM/SETOP. Volume 8147 of LNCS. (2013) 34–50
4. Barbosa, M., Brouard, T., Cauchie, S., de Sousa, S.M.: Secure biometric authentication with improved accuracy. In: ACISP 2008. Volume 5107 of LNCS., Springer (2008) 21–36
5. Erkin, Z., Franz, M., Guajardo, J., Katzenbeisser, S., Lagendijk, I., Toft, T.: Privacy-preserving face recognition. In: PETS 2009. (2009) 235–253
6. Sadeghi, A.R., Schneider, T., Wehrenberg, I.: Efficient privacy-preserving face recognition. In: ICISC 2009. LNCS (2009) 229–244
7. Huang, Y., Malka, L., Evans, D., Katz, J.: Efficient privacy-preserving biometric identification. In: NDSS. (2011)
8. Abidin, A., Mitrokotsa, A.: Security aspects of privacy-preserving biometric authentication based on ideal lattices and ring-lwe. In: Proceedings of the IEEE Workshop on Information Forensics and Security. (2014) 1653–1658
9. Gennaro, R., Gentry, C., Parno, B.: Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In: CRYPTO 2010. LNCS (2010) 465–482
10. Chung, K.M., Kalai, Y.T., Vadhan, S.P.: Improved delegation of computation using fully homomorphic encryption. In: CRYPTO 2010. LNCS (2010) 483–501
11. Benabbas, S., Gennaro, R., Vahlis, Y.: Verifiable delegation of computation over large datasets. In: CRYPTO 2011. Volume 6841 of LNCS. (2011) 111–131
12. Backes, M., Fiore, D., Reischuk, R.M.: Verifiable delegation of computation on outsourced data. In: ACM CCS 2013, ACM (2013) 863–874
13. Setty, S.T., McPherson, R., Blumberg, A.J., Walfish, M.: Making argument systems for outsourced computation practical (sometimes). In: NDSS 2012. (2012)
14. Zhang, L.F., Safavi-Naini, R.: Batch verifiable computation of outsourced functions. Designs, Codes and Cryptography (2015) 1–23
15. Papamanthou, C., Shi, E., Tamassia, R.: Signatures of correct computation. In: TCC 2013. LNCS (2013) 222–242
16. Rabin, M.O.: How to exchange secrets with oblivious transfer. IACR Cryptology ePrint Archive **2005** (2005) 187

17. Yao, A.C.C.: How to generate and exchange secrets. In: Foundations of Computer Science, 1986., 27th Annual Symposium on, IEEE (1986) 162–167
18. Chor, B., Kushilevitz, E., Goldreich, O., Sudan, M.: Private information retrieval. Journal of the ACM **45**(6) (1998) 965–981
19. Ostrovsky, R., Willian E. Skeith, I.: A survey of single-database private information retrieval: techniques and applications. In: PKC'07. LNCS, Springer (2007) 393–411
20. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: EUROCRYPT 1999. Volume 1592 of LNCS. (1999) 223–238
21. Goldwasser, S., Micali, S.: Probabilistic encryption & how to play mental poker keeping secret all partial information. In: Proceedings of the fourteenth annual ACM symposium on Theory of computing. STOC 1982, ACM (1982) 365–377
22. Damgård, I., Geisler, M., Krøigaard: Efficient and secure comparison for on-line auctions. In: ACISP 2007. Volume 4586 of LNCS., Springer (2007) 416–430
23. Osadchy, M., Pinkas, B., Jarrous, A., Moskovich, B.: SCiFI - A System for Secure Face Identification. In: IEEE S&P 2010. (May 2010) 239–254
24. Simoens, K., Bringer, J., Chabanne, H., Seys, S.: A framework for analyzing template security and privacy in biometric authentication systems. IEEE Transactions on Information Forensics and Security **7**(2) (2012) 833–841
25. Abidin, A., Matsuura, K., Mitrokotsa, A.: Security of a privacy-preserving biometric authentication protocol revisited. In: CANS 2014. Volume 8813 of LNCS., Springer (2014) 290–304
26. Abidin, A., Pagnin, E., Mitrokotsa, A.: Attacks on privacy-preserving biometric authentication. In: Proceedings of the 19th Nordic Conference on Secure IT Systems (NordSec 2014). LNCS, Springer (October 2014) 293–294
27. Bringer, J., Chabanne, H., Patey, A.: SHADE: Secure hamming distance computation from oblivious transfer. In: Financial Cryptography Workshops. (2013) 164–176
28. Bringer, J., Chabanne, H., Favre, M., Patey, A., Schneider, T., Zohner, M.: GSHADE: Faster Privacy-preserving Distance Computation and Biometric Identification. In: Proceedings of the 2nd ACM Workshop on Information Hiding and Multimedia Security, ACM (2014) 187–198
29. Pagnin, E., Dimitrakakis, C., Abidin, A., Mitrokotsa, A.: On the leakage of information in biometric authentication. In: INDOCRYPT 2014. LNCS, Springer (2014) 265–280
30. Krawczyk, H.: Lfsr-based hashing and authentication. In Desmedt, Y., ed.: CRYPTO '94. Volume 839 of Lecture Notes in Computer Science., Springer 1994 (1994) 129–139
31. Carter, L., Wegman, M.N.: Universal classes of hash functions. J. Comput. Syst. Sci. **18** (1979) 143–154
32. Stinson, D.R.: Universal hashing and authentication codes. In Feigenbaum, J., ed.: CRYPTO '91. Volume 576 of Lecture Notes in Computer Science., Springer 1992 (1991) 74–85
33. Abidin, A., Larsson, J.Å.: New universal hash functions. In Lucks, S., Armknecht, F., eds.: WEWoRC 2011. Volume 7242 of LNCS., Springer-Verlag (2012) 99–108

A Proof of Theorem 1

Before we proceed with the proof, let us first analyse the adversarial scenario in the case of the generic protocol PPBA. Note that by the

attacker (or the adversary) $\mathcal{A}$, we refer to the malicious cloud server. We assume that the adversary $\mathcal{A}$ has oracle access to Authen, so $\mathcal{A}$ can query Authen with biometric templates and identity of its choice $\mathsf{poly}(\lambda)$ times, where λ is a security parameter. In addition, by the security of a privacy-preserving biometric authentication protocol, we mean the security of the biometric templates.

Again, we define the security of the protocol PPBA against a malicious adversary $\mathcal{A}$ via the following game played between $\mathcal{A}$ and PPBA.

$\mathsf{Exp}^{\mathsf{Priv}}_{\mathsf{PPBA},\mathcal{A}}(\lambda, f)$:
$(\mathsf{pk}, \mathsf{sk})$, $(\mathsf{PK}, \mathsf{VK}) \leftarrow \mathsf{KeyGen}(\lambda, f)$
$(\mathsf{ID}_i, b'_{i_0}, b'_{i_1})$, $b'_{i_0} \neq b'_{i_1} \leftarrow \mathcal{A}(\lambda, \mathsf{pk}, \mathsf{PK}, f)$
$\beta \xleftarrow{R} \{0,1\}$; $\mathsf{Out} \leftarrow \mathsf{Authen}\big(\mathsf{ID}_i, i, \mathsf{Enc}(b'_{i_\beta})\big)$
$\beta' \leftarrow \mathcal{A}\big(\mathsf{ID}_i, b'_{i_0}, b'_{i_1}, \mathsf{Enc}(b'_{i_\beta}), \mathsf{Out}\big)$
Return 1 if $\beta' = \beta$, 0 otherwise

The adversary's advantage at the end of this game is defined as $\mathsf{Adv}^{\mathsf{Priv}}_{\mathsf{PPBA},\mathcal{A}} = \big|2\Pr\{\mathsf{Exp}^{\mathsf{Priv}}_{\mathsf{PPBA},\mathcal{A}}(\lambda, f) = 1\} - 1\big|$. We say that the protocol is secure (and preserves the privacy of biometric templates) against the malicious cloud server $\mathcal{CS}$, if $\mathsf{Adv}^{\mathsf{Priv}}_{\mathsf{PPBA},\mathcal{A}} \leqslant \mathsf{negl}(\lambda)$.

Let us write out the details of $\mathsf{Authen}\big(\mathsf{ID}_i, i, \mathsf{Enc}(b'_{i_\beta})\big)$ in the above experiment. Since the authentication process involves the client $\mathcal{C}_i$, the cloud server $\mathcal{CS}$, and the service provider $\mathcal{SP}$, in the description we write the entity name followed by a set of inputs it takes in a parenthesis to denote what that entity takes as input. For instance, $\mathcal{CS}(i, \mathsf{Enc}(b'_{i_\beta}), \mathsf{pk}, \mathsf{PK})$ denotes that $\mathcal{CS}$ takes i, $\mathsf{Enc}(b'_{i_\beta})$, and PK as input and performs the operations in the indented block underneath it.

$\mathsf{Authen}\big(\mathsf{ID}_i, i, \mathsf{Enc}(b'_{i_\beta})\big)$:

$\mathcal{C}_i$:
  Send $(i, \mathsf{Enc}(b'_{i_\beta}))$ to $\mathcal{CS}$
$\mathcal{CS}(i, \mathsf{Enc}(b'_{i_\beta}), \mathsf{pk}, \mathsf{PK})$:
  $\mathsf{Enc}(b_i) \leftarrow \mathcal{DB}(i)$
  $\sigma_{\mathsf{ct}_i} \leftarrow \mathsf{Com}(\mathsf{Enc}(b_i), \mathsf{Enc}(b'_{i_\beta}), \mathsf{pk}, \mathsf{PK})$
  Send $(\mathsf{Enc}(b_i), \sigma_{\mathsf{ct}_i})$ to $\mathcal{C}_i$
$\mathcal{C}_i\big(\mathsf{Enc}(b'_{i_\beta}), \mathsf{Enc}(b_i), \sigma_{\mathsf{ct}_i}\big)$:
  $\mathsf{ct}_i \leftarrow \mathsf{Ver}\big(\mathsf{Enc}(b_i), \mathsf{Enc}(b'_{i_\beta}), \sigma_{\mathsf{ct}_i}, \mathsf{VK}\big)$
  if $\mathsf{ct}_i = \perp$ then
    Return $\boxed{\mathsf{Out}=0}$
  else
    $\widetilde{\omega}_i \leftarrow H(\mathsf{Enc}(b_i))$
    Send $(\mathsf{ID}_i, \mathsf{ct}_i, \widetilde{\omega}_i)$ to $\mathcal{SP}$

$\mathcal{SP}(\mathsf{ID}_i, \mathsf{ct}_i, \widetilde{\omega}_i, \mathsf{sk})$:
  $i \leftarrow \mathsf{ID}_i$
  $\omega_i \leftarrow \mathsf{db}(i)$
  if $\widetilde{\omega}_i \neq \omega_i$ then
    Return $\boxed{\mathsf{Out}=0}$
  else
    $\mathsf{Dist} \leftarrow \mathsf{Dec}(\mathsf{ct}_i)$
    if $\mathsf{Dist} \leqslant \tau$ then
      Return $\boxed{\mathsf{Out}=1}$
    else
      Return $\boxed{\mathsf{Out}=0}$

In the authentication process Authen, $\mathsf{Out} = 1$ is returned in only one case (i.e., the case where the fresh and the reference biometric templates match each other), while $\mathsf{Out} = 0$ is returned in three cases. The three cases are (1) $\mathcal{CS}$ does not perform the correct computation and the verification algorithm Ver outputs $\perp$, (2) $\mathcal{CS}$ performs the correct computation but uses a wrong input, so the integrity check fails, finally (3) there is no match between the fresh and the reference biometric templates.

Proof (of Theorem 1). We prove this theorem using two games.
game 0: This is the original game. Let S_0 be the event that $\beta' = \beta$.
game 1: This is the same as **game** 0, except that we now replace the output $(\mathsf{Enc}(b_i), \sigma_{\mathsf{ct}_i}) \leftarrow \mathcal{CS}(i, \mathsf{Enc}(b'_{i_\beta}), \mathsf{pk}, \mathsf{PK})$ with the correct $\mathsf{Enc}(b_i)$ corresponding to i and valid σ_{ct_i}. Let S_1 be the event that $\beta' = \beta$ in this game.
Claim 1: $|\Pr\{S_0\} - \Pr\{S_1\}|$ is negligible.

*Proof (of **Claim** 1).* The difference between **game** 0 and **game** 1 is that in **game** 0 it may happen that $\mathsf{ct}_i = \perp$ and/or $\widetilde{\omega}_i \neq \omega_i$, while in **game** 1 these do not happen. While $\mathsf{ct}_i = \perp$ means winning the game $\mathsf{Exp}_{\mathsf{VC},\mathcal{A}}(\lambda, f)$, $\widetilde{\omega}_i \neq \omega_i$ means having a collision in H. So both of these happen with negligible probability because of the assumption that VC is secure (cf. Definition 7) and that H is a random oracle. Therefore, the difference between the winning probabilities in **game** 0 and **game** 1 is negligible.

Claim 2: $\left|2\Pr\{S_1\} - 1\right| \leqslant \mathsf{negl}(\lambda)$.

*Proof (of **Claim** 2).* Suppose that the adversary's advantage is non-negligible, i.e., $\left|2\Pr\{S_1\} - 1\right| > \mathsf{negl}(\lambda)$. Then we can construct an attacker $\mathcal{A}'$ that wins in the $\mathsf{IND\text{-}CPA}$ game against the underlying homomorphic encryption HE with non-negligible advantage as follows.

$\mathsf{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathsf{HE},\mathcal{A}}(\lambda)$:
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\lambda)$; $(m_0, m_1), m_0 \neq m_1 \leftarrow \mathcal{A}'(\lambda, \mathsf{pk})$
$\alpha \xleftarrow{R} \{0,1\}$; $c \leftarrow \mathsf{Enc}(m_\alpha, \mathsf{pk})$; Simulate PPBA for $\mathcal{A}$
$\alpha' (= \beta') \leftarrow \mathcal{A}'(\mathcal{A}(m_0, m_1, c, \mathsf{pk}))$
Return 1 if $\alpha' = \alpha$, 0 otherwise

The attacker $\mathcal{A}'$ obtains the pk for HE, chooses two distinct messages $m_0, m_1 \in \mathbb{Z}^N_{q \geqslant 2}$, and receives a challenge $c = \mathsf{Enc}(m_\alpha)$, where $\alpha \xleftarrow{R} \{0,1\}$. $\mathcal{A}'$ then simulates the protocol execution for PPBA. To simulate PPBA, $\mathcal{A}'$ uses pk to re-randomise $c = \mathsf{Enc}(m_\alpha)$ using the homomorphic property of the encryption, and registers the re-randomised c, let us call it c', along with an ID_i and a corresponding index i and a hash of c' in $\mathcal{DB}$ of $\mathcal{CS}$.

For $\mathcal{CS}$, c and its randomised version c' are indistinguishable. This does faithfully simulate the protocol execution for the adversary $\mathcal{A}$, because $\mathcal{A}'$ knows the output of $\mathsf{Authen}(\mathsf{ID}_i, i, c)$. Now, if $\mathcal{A}$ outputs its guess β' for β, then $\mathcal{A}'$ outputs its guess $\alpha'(=\beta')$ for α. Thus, $\mathcal{A}'$ wins if $\mathcal{A}$ wins.

Hence, combining **Claim** 1 and 2, we have that $\mathsf{Adv}^{\mathsf{Priv}}_{\mathsf{PPBA},\mathcal{A}}$ is negligible.

B Universal hash functions

Universal hash functions were first proposed by Carter and Wegman [31] as, among others, a means to construct unconditionally secure MACs. Stinson formalised the definitions of Universal hash functions in [32]. Following these early works, there has been a considerable amount of research done on Universal hash functions to improve both the description length and computational performance, see e.g., [33] for a quick overview.

Definition 9 (ϵ-ASU$_2$ hash functions [32]). *Let $\mathcal{M}$ and $\mathcal{T}$ be finite sets. A family $\mathcal{F}$ of hash functions from $\mathcal{M}$ to $\mathcal{T}$ is ϵ-ASU$_2$ if the following two conditions are satisfied: (a) the number of hash functions in $\mathcal{F}$ that takes an arbitrary $m_1 \in \mathcal{M}$ to an arbitrary $t_1 \in \mathcal{T}$ is exactly $|\mathcal{F}|/|\mathcal{T}|$; (b) the fraction of those functions that also takes an arbitrary $m_2 \neq m_1$ in $\mathcal{M}$ to an arbitrary $t_2 \in \mathcal{T}$ (possibly equal to t_1) is at most ϵ. If $\epsilon = 1/|\mathcal{T}|$, then $\mathcal{F}$ is called* SU$_2$.

As can be seen from the definition, ϵ-ASU$_2$ hash functions can be used to construct a MAC scheme in a natural way. More specifically, in this case a pair of users, say Alice and Bob, share a secret key k which identifies a hash function h_{k} in a family of ϵ-ASU$_2$ hash functions. When Alice sends a message m to Bob, she also sends $t = h_{\mathsf{k}}(m)$ along with m. Upon receiving (m, t), Bob checks the authenticity of m by comparing t with $h_{\mathsf{k}}(m)$, which he himself computes using his share of the key k. If $h_{\mathsf{k}}(m) = t$, then Bob accepts m as authentic; otherwise, he rejects it.

C Proof of Theorem 2

Proof (of Theorem 2). Since the proof is similar to that of the Theorem 1, we just highlight the differences in the relevant hybrid security games and the claims. Let PPBA-HE-MAC denote the instantiation. The security against a malicious adversary $\mathcal{A}$ (e.g., $\mathcal{CS}$) is defined via the following game played between $\mathcal{A}$ and PPBA-HE-MAC.

$\mathsf{Exp}^{\mathsf{Priv}}_{\mathsf{PPBA\text{-}HE\text{-}MAC},\mathcal{A}}(\lambda)$:
$(\mathsf{pk},\mathsf{sk})$, $\mathsf{MAC}.K \leftarrow \mathsf{KeyGen}(\lambda)$
$(\mathsf{ID}_i, b'_{i_0}, b'_{i_1})$, $b'_{i_0} \neq b'_{i_1} \leftarrow \mathcal{A}(\lambda, \mathsf{pk}, \mathsf{MAC}.K)$
$\beta \xleftarrow{R} \{0,1\}$; $\mathsf{Out} \leftarrow \mathsf{Authen}\big(\mathsf{ID}_i, i, \mathsf{Enc}(b'_{i_\beta})\big)$
$\beta' \leftarrow \mathcal{A}\big(\mathsf{ID}_i, b'_{i_0}, b'_{i_1}, \mathsf{Enc}(b'_{i_\beta}), \mathsf{Out}\big)$
Return 1 if $\beta' = \beta$, 0 otherwise

where $\mathsf{MAC}.K$ is the key space for the employed MAC scheme (e.g., the set of U_2 hash functions). The adversary's advantage is defined as $\mathsf{Adv}^{\mathsf{Priv}}_{\mathsf{PPBA\text{-}HE\text{-}MAC},\mathcal{A}} = \big|2\Pr\{\mathsf{Exp}^{\mathsf{Priv}}_{\mathsf{PPBA\text{-}HE\text{-}MAC},\mathcal{A}}(\lambda) = 1\} - 1\big|$. If $\mathsf{Adv}^{\mathsf{Priv}}_{\mathsf{PPBA\text{-}HE\text{-}MAC},\mathcal{A}} \leqslant \mathsf{negl}(\lambda)$, we say that PPBA-HE-MAC is secure (and preserves the privacy of biometric templates) against $\mathcal{A}$.

The details of $\mathsf{Authen}\big(\mathsf{ID}_i, i, \mathsf{Enc}(b'_{i_\beta})\big)$ are given below.

Authen$\big(\mathsf{ID}_i, i, \mathsf{Enc}(b'_{i_\beta})\big)$**:**
$\mathcal{C}_i$ sends $(i, \mathsf{Enc}(b'_{i_\beta}))$ to $\mathcal{CS}$
$\mathcal{C}_i$ sends $(\mathsf{ID}_i, t'_{i_\beta})$ to $\mathcal{SP}$
$\mathcal{CS}(i, \mathsf{Enc}(b'_{i_\beta}), \mathsf{pk})$:
$\mathsf{Enc}(b_i) \leftarrow \mathcal{DB}(i)$
$\gamma_i \leftarrow \mathsf{Enc}(b_i)\mathsf{Enc}(b'_{i_\beta}) = \mathsf{Enc}(b_i \oplus b'_{i_\beta})$
Send (i, γ_i) to $\mathcal{SP}$
$\mathcal{SP}(\mathsf{ID}_i, i, \gamma_i, t'_{i_\beta}, \mathsf{sk})$:
If i is not the correct index for ID_i then
Return $\boxed{\mathsf{Out}{=}0}$
$(\mathsf{k}_i, t_i) \leftarrow \mathsf{db}(i)$
if $t_i \oplus t'_{i_\beta} \neq \mathsf{TAG}(\mathsf{Dec}(\gamma_i), \mathsf{k}_i)$ then
Return $\boxed{\mathsf{Out}{=}0}$
else
if $\mathsf{HW}(\gamma_i) \leqslant \tau$ then
Return $\boxed{\mathsf{Out}{=}1}$
else
Return $\boxed{\mathsf{Out}{=}0}$

The proof is based on the following two hybrid games.
game 0: This is the original game $\mathsf{Exp}^{\mathsf{Priv}}_{\mathsf{PPBA\text{-}HE\text{-}MAC},\mathcal{A}}(\lambda)$. Let S_0 be the event that $\beta' = \beta$ in **game** 0.
game 1: This is the same as **game** 0, except that now $\mathcal{CS}$ always performs the correct computation. Let S_1 be the event that $\beta' = \beta$ in **game** 1.
Claim 1: $|\Pr\{S_0\} - \Pr\{S_1\}|$ is negligible. This follows from the ϵ-security of the employed MAC scheme.
Claim 2: The adversary has negligible advantage in **game** 1, i.e., $\big|2\Pr\{S_1\} - 1\big| \leqslant \mathsf{negl}(\lambda)$. This follows from the IND-CPA-security of the HE scheme.

Hence, we have that $\mathsf{Adv}^{\mathsf{Priv}}_{\mathsf{PPBA\text{-}HE\text{-}MAC},\mathcal{A}}$ is negligible.